Understanding How The Bad Guys Attack Your Software: CAPEC

Sean Barnum
MITRE
Our Reality

■ Security issues are becoming increasingly critical to organizations developing software

■ Today’s software developers must consider security as a requisite property of software they develop and testing and quality assurance (QA) professionals are increasingly being asked to take on verification and testing of security issues in addition to traditional quality issues

■ For people unfamiliar with security this can appear a daunting task and requires resources to help in this transition
Traditional Testing

- Test strategy
- Test planning from existing requirements
- Test case design
- Test automation
- Test execution
- Defect tracking and change management
Traditional Quality Assurance

- Requirements Reviews
- Design Reviews
- Code Reviews
- Traditional Testing
Extending Traditional QA to Include Security

- Security Requirements Capture and Analysis including Abuse Cases
- Architectural Risk Analysis
- Secure Code Review
- Risk-based Security Testing
- Penetration Testing
What New Dimensions does Security Bring?

- Don’t stop what you are doing, just build on it
- Evidence that software does what it is supposed to do and nothing else
- Intentional vs Unintentional problems
- Testing Inside-Out not just Outside-In
- Risk-based approach
  - Software will never be perfect
  - Valid and valuable for QA
  - Crucial for security
- Recognize the attacker’s perspective
  - Think like the bad guys
The Long-established Principal of “Know Your Enemy”

“One who knows the enemy and knows himself will not be endangered in a hundred engagements. One who does not know the enemy but knows himself will sometimes be victorious. Sometimes meet with defeat. One who knows neither the enemy nor himself will invariably be defeated in every engagement.”

Chapter 3: “Planning the Attack”
The Art of War, Sun Tzu
The Importance of Knowing Your Enemy

- An appropriate defense can only be established if you know how it will be attacked

- Remember!
  - Software Assurance must assume motivated attackers and not simply passive quality issues
  - Attackers are very creative and have powerful tools at their disposal
  - Exploring the attacker’s perspective helps to identify and qualify the risk profile of the software
What are Attack Patterns?

- Blueprint for creating a specific type of attack
- Abstracted common attack approaches from the set of known exploits
- Capture the attacker’s perspective to aid software developers, acquirers and operators in improving the assurance profile of their software
Leveraging Attack Patterns Throughout the Software Lifecycle

- Guide definition of appropriate policies
- Guide creation of appropriate security requirements (positive and negative)
- Provide context for architectural risk analysis
- Guide risk-driven secure code review
- Provide context for appropriate security testing
- Provide a bridge between secure development and secure operations
Common Attack Pattern Enumeration and Classification (CAPEC)

- **Community effort targeted at:**
  - Standardizing the capture and description of attack patterns
  - Collecting known attack patterns into an integrated enumeration that can be consistently and effectively leveraged by the community
  - Gives you an attacker’s perspective you may not have on your own

- **Excellent resource for many key activities**
  - Abuse Case development
  - Architecture attack resistance analysis
  - Risk-based security/Red team penetration testing
  - Whitebox and Blackbox testing correlation
  - Operational observation and correlation

- **Where is CAPEC today?**
  - [http://capec.mitre.org](http://capec.mitre.org)
  - Currently 386 patterns, stubs, named attacks
The HS SEDI FFRDC is managed and operated by The MITRE Corporation for DHS.

Common Attack Pattern Enumeration and Classification
A Community Knowledge Resource for Building Secure Software

Building software with an adequate level of security assurance for its mission becomes more and more challenging every day as the size, complexity, and tempo of software creation increases and the number and the skill level of attackers continues to grow. These factors each exacerbate the issue that, to build secure software, builders must ensure that they have protected every relevant potential vulnerability; yet, to attack software, attackers often have to find and exploit only a single exposed vulnerability. To identify and mitigate relevant vulnerabilities in software, the development community needs more than just good software engineering and analytical practices, a solid grasp of software security features, and a powerful set of tools. All of these things are necessary but not sufficient. To be effective, the community needs to think outside of the box and to have a firm grasp of the attacker’s perspective and the approaches used to exploit software.

Attack patterns are a powerful mechanism to capture and communicate the attacker’s perspective. They are descriptions of common methods for exploiting software. They derive from the concept of design patterns applied in a destructive rather than constructive context and are generated from in-depth analysis of specific real-world exploit examples.

To assist in enhancing security throughout the software development lifecycle, and to support the needs of developers, testers and educators, the Common Attack Pattern Enumeration and Classification (CAPEC) is sponsored by the Department of Homeland Security as part of the Software Assurance strategic initiative of the National Cyber Security Division. The objective of this effort is to provide a publicly available catalog of attack patterns along with a comprehensive schema and classification taxonomy. This site now contains the initial set of content and will continue to evolve with public participation and contributions to form a standard mechanism for identifying, collecting, refining, and sharing attack patterns among the software community.

Release 1.6 Available

Page Last Updated: March 11, 2011

CAPEC is a Software Assurance strategic initiative co-sponsored by the National Cyber Security Division of the U.S. Department of Homeland Security. This Web site is sponsored and managed by The MITRE Corporation in enable stakeholder collaboration. Copyright 2011 The MITRE Corporation. CAPEC and the CAPEC logo are trademarks of The MITRE Corporation. Contact capemitre.org for more information.
What do Attack Patterns Look Like?

■ Primary Schema Elements
  – Identifying Information
    ■ Attack Pattern ID
    ■ Attack Pattern Name
  – Describing Information
    ■ Description
    ■ Related Weaknesses
    ■ Related Vulnerabilities
    ■ Method of Attack
    ■ Examples-Instances
    ■ References
  – Prescribing Information
    ■ Solutions and Mitigations
  – Scoping and Delimiting Information
    ■ Typical Severity
    ■ Typical Likelihood of Exploit
    ■ Attack Prerequisites
    ■ Attacker Skill or Knowledge Required
    ■ Resources Required
    ■ Attack Motivation-Consequences
    ■ Context Description

■ Supporting Schema Elements
  – Describing Information
    ■ Injection Vector
    ■ Payload
    ■ Activation Zone
    ■ Payload Activation Impact
  – Diagnosing Information
    ■ Probing Techniques
    ■ Indicators-Warnings of Attack
    ■ Obfuscation Techniques
  – Enhancing Information
    ■ Related Attack Patterns
    ■ Relevant Security Requirements
    ■ Relevant Design Patterns
    ■ Relevant Security Patterns
Attack Pattern Description Schema Formalization

Description

■ Summary

■ Attack_Execution_Flow
  – Attack_Phase\(^{1..3}\) (Name(Explore, Experiment, Exploit))
    ■ Attack_Step\(^{1..*}\)
      - Attack_Step_Title
      - Attack_Step_Description
      - Attack_Step_Technique \(^{0..*}\)
        ■ Attack_Step_Technique_Description
        ■ Leveraged_Attack_Patterns
        ■ Relevant_Attack_Surface_Elements
        ■ Observables\(^{0..*}\)
        ■ Environments
      - Indicator\(^{0..*}\) (ID, Type(Positive, Failure, Inconclusive))
        ■ Indicator_Description
        ■ Relevant_Attack_Surface_Elements
        ■ Environments
      - Outcome\(^{0..*}\) (ID, Type(Success, Failure, Inconclusive))
        ■ Outcome_Description
        ■ Relevant_Attack_Surface_Elements
        ■ Observables\(^{0..*}\)
        ■ Environments
      - Security Control\(^{0..*}\) (ID, Type(Detective, Corrective, Preventative))
        ■ Security_Control_Description
        ■ Relevant_Attack_Surface_Elements
        ■ Observables\(^{0..*}\)
        ■ Environments
  ■ Leveraged_Attack_Patterns
  ■ Relevant_Attack_Surface_Elements
  ■ Observables\(^{0..*}\)
  ■ Environments
Blind SQL Injection

Summary

Blind SQL Injection results from an insufficient mitigation for SQL Injection. Although suppressing database error messages are considered best practice, the suppression alone is not sufficient to prevent SQL Injection. Blind SQL Injection is a form of SQL Injection that overcomes the lack of error messages. Without the error messages that facilitate SQL Injection, the attacker constructs input strings that probe the target through simple Boolean SQL expressions. The attacker can determine if the syntax and structure of the injection was successful based on whether the query was executed or not. Applied iteratively, the attacker determines how and where the target is vulnerable to SQL Injection.

In order to achieve this using Blind SQL Injection, an attacker:

For example, an attacker may try entering something like "username' AND 1=1; --" in an input field. If the result is the same as when the attacker entered "username" in the field, then the attacker knows that the application is vulnerable to SQL Injection. The attacker can then ask yes/no questions from the database server to extract information from it. For example, the attacker can extract table names from a database using the following types of queries:

"username' AND ascii(lower(substring(((SELECT TOP 1 name FROM sysobjects WHERE xtype='U'), 1, 1))) > 108".

If the above query executes properly, then the attacker knows that the first character in a table name in the database is a letter between m and z. If it doesn't, then the attacker knows that the character must be between a and I (assuming of course that table names only contain alphabetic characters). By performing a binary search on all character positions, the attacker can determine all table names in the database. Subsequently, the attacker may execute an actual attack and send something like:

"username'; DROP TABLE trades; --"
Complete CAPEC Entry Information
A Few Key Use Cases for CAPEC in Support of SwA

- Help developers understand weaknesses in their real-world context (how they will be attacked)
- Objectively identify specific attacks under which software must demonstrate resistance, tolerance and resilience for a given level of assurance
- Indirectly scope which weaknesses are relevant for a given threat environment
- Identify relevant mitigations that should be applied as part of policy, requirements, A&D, implementation, test, deployment and operations
- Identify and characterize patterns of attacks for security test case generation
- Identify and characterize threat TTPs for red teaming
- Identify relevant issues for automated tool selection
- Identify and characterize issues for automated tool results analysis
CAPEC and Security Measurement

- Measuring stick for evaluating and comparing penetration testing tools and application defense tools
  ■ Similar to CWE value to secure code analysis tools and CVE value to vulnerability scanners

- Measuring stick for attack resistance claims of assurance cases
  ■ It comes down to the proxies for measuring security (vulnerabilities, weaknesses and attack resistance)

- Characterizing the nature of software attack
  ■ Formalization of attack patterns to enable:
    - Recognition and mapping of attack instances from the operations realm
    - Refined ability to measure attack resistance in terms of resistance to individual sub-elements of attack with observables
    - Automated generation of penetration attack cases
    - Defining & mapping attack simulations in penetration testing tools
  ■ Alignment with malware characterization (MAEC)
CAPEC Status

Where is CAPEC today?

• V1.4
  • Massive schema changes
    • Including addition of Observables structure
  • Some new content
  • Added initial set of network attack patterns

• V1.5
  • Added ~25 new network attack patterns
  • Added enhanced material to ~35 patterns
  • New View added for WASC Threat Taxonomy 2.0
  • Added ~65 mappings to CWE and several within CAPEC

• V1.6
  • Added 7 new application framework attack patterns as well as 68 new attack patterns in three new attack pattern categories: Physical Security Attacks, Social Engineering Attacks & Supply Chain Attacks
  • Added ~35 mappings to CWE and several within CAPEC

Currently 386 patterns, stubs, named attacks; 68 categories and 6 views
CAPEC Current Content (15 Major Categories)

1000 - Mechanism of Attack
• Data Leakage Attacks - (118)
• Resource Depletion - (119)
• Injection (Injecting Control Plane content through the Data Plane) - (152)
• Spoofing - (156)
• Time and State Attacks - (172)
• Abuse of Functionality - (210)
• Exploitation of Authentication - (225)
• Probabilistic Techniques - (223)
• Exploitation of Privilege/Trust - (232)
• Data Structure Attacks - (255)
• Resource Manipulation - (262)
• Physical Security Attacks (436)
• Network Reconnaissance - (286)
• Social Engineering Attacks (403)
• Supply Chain Attacks (437)
CAPEC Current Content (Which Expand to...)

1000 - Mechanism of Attack

Data Leakage Attacks - (118)
- Data Excavation Attacks - (116)
- Data Interception Attacks - (117)

Resource Depletion - (119)
- Violating Implicit Assumptions Regarding XML Content (aka XML Denial Exploitation of Privilege/Trust) - (232)
- Data Depletion through Flooding - (125)
- Resource Depletion through Allocation - (130)
- Resource Depletion through Leak - (131)
- Denial of Service through Resource Depletion - (227)

Injection (Injecting Control Plane content through the Data Plane) - (152)
- Remote Code Inclusion - (253)
  - Analog In-band Switching Signals (aka Blue Boxing) - (5)
  - SQL Injection - (66)
  - Email Injection - (134)
  - Format String Injection - (135)
  - LDAP Injection - (136)
  - Parameter Injection - (137)
  - Reflection Injection - (138)
  - Code Inclusion - (175)
  - Resource Injection - (240)
  - Script Injection - (242)
  - Command Injection - (248)
  - Character Injection - (249)
  - XML Injection - (250)
  - DTD Injection in a SOAP Message - (254)

Spoofing - (156)
- Content Spoofing - (148)
  - Identity Spoofing (Impersonation) - (151)
  - Action Spoofing - (173)

Time and State Attacks - (172)
- Forced Deadlock - (25)
- Leveraging Race Conditions - (26)
- Leveraging Time-of-Check and Time-of-Use (TOCTOU) Race Conditions - (29)
- Manipulating User State - (74)

Abuse of Functionality - (210)
- Functionality Misuse - (212)
- Abuse of Communication Channels - (216)
- Forceful Browsing - (87)
- Passing Local Filenames to Functions That Expect a URL - (48)
- Probing an Application Through Targeting its Error Reporting - (54)
- WSDL Scanning - (95)
- API Abuse/Misuse - (113)
- Try All Common Application Switches and Options - (133)
- Cache Poisoning - (141)
- Software Integrity Attacks - (184)
- Directory Traversal - (213)

Analytic Attacks - (281)

Probabilistic Techniques - (223)

Exploitation of Authentication - (225)
- Exploitation of Session Variables, Resource IDs and other Trusted Credentials - (21)
- Authentication Abuses - (114)
- Exploitation of Privilege/Trust - (232)
- Privilege Escalation - (233)
- Exploiting Trust in Client (aka Make the Client Invisible) - (22)
  - Hijacking a Privileged Thread of Execution - (30)
  - Subvert Code-signing Facilities - (88)
  - Target Programs with Elevated Privileges - (69)
  - Exploitation of Authorization - (122)

Resource Manipulation - (262)
- Accessing/Intercepting/Modifying HTTP Cookies - (31)
- Buffer Attacks - (123)
- Attack through Shared Data - (124)
- Integer Attacks - (128)
- Pointer Attack - (129)

Network Reconnaissance - (286)
- ICMP Echo Request Ping - (285)
- TCP SYN Scan - (287)
- ICMP Echo Request Ping - (288)
- Infrastructure-based footprinting - (299)
- Enumerate Mail Exchange (MX) Records - (290)
- DNS Zone Transfers - (291)
- Host Discovery - (292)
- Traceroute Route Enumeration - (293)
- ICMP Address Mask Request - (294)
- ICMP Timestamp Request - (295)
- ICMP Information Request - (296)
- TCP ACK Ping - (297)
- UDP Ping - (298)
- TCP SYN Ping - (299)
- Port Scanning - (300)
- TCP Connect Scan - (301)
- TCP FIN scan - (302)
- TCP Xmas Scan - (303)
- TCP Null Scan - (304)

TCP ACK Scan - (305)
TCP Window Scan - (306)
TCP RPC Scan - (307)
UDP Scan - (308)
CAPEC Current Content (386 Attacks...)
Current Maturation Paths

- Extend coverage of CAPEC
- Improve quality of CAPEC
- Expand the scope of CAPEC
- Improve integration with other standards (MAEC, CEE, etc.)
- Expand use of CAPEC
- Bridge secure development with secure operations
## Attack Patterns Help Answer Foundational Questions Regarding Secure Operations

<table>
<thead>
<tr>
<th>Question</th>
<th>Role of Attack Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Are we being attacked? (Were we attacked?)</td>
<td>Attack patterns offer structured descriptions of common attacker behaviors to help interpret observed operational data and determine its innocent or malicious intent.</td>
</tr>
<tr>
<td>How are we being attacked?</td>
<td>Attack patterns offer detailed structured descriptions of common attacker behavior to help interpret observed operational data and determine exactly what sort of attack is occurring.</td>
</tr>
<tr>
<td>What is the objective of the attack?</td>
<td>Elements of attack patterns outlining attacker motivation and potential attack effects can be leveraged to help map observed attack behaviors to potential attacker intent.</td>
</tr>
<tr>
<td>What is our exposure?</td>
<td>The structure detail and weakness mapping of attack patterns can provide guidance in where to look and what to look for when certain attack pattern behaviors are observed.</td>
</tr>
<tr>
<td>Who is attacking us?</td>
<td>Attack pattern threat characterization and detailed attack execution flow can provide a framework for organizing real-world attack data to assist in attribution.</td>
</tr>
<tr>
<td>What should we do to prevent against attacks in the future?</td>
<td>Attack patterns offer prescriptive guidance on solutions and mitigation approaches that can be effective in improving the resistance tolerance and/or resilience to instances of a given pattern of attack.</td>
</tr>
</tbody>
</table>
So, this all sounds great but how do we map these high-level attack pattern abstractions to the low-level operational world?

Cyber Observables

The Secret Sauce for Bridging the Abstract to the Concrete
What is a cyber observable?

- a **measurable event** or **stateful property** in the cyber domain
  
  - Some measurable events: a registry key is created, a file is deleted, an http GET is received, …
  
  - Some stateful properties: MD5 hash of a file, value of a registry key, existence of a mutex, …

Cyber Observable eXpression (CybOX) is a standardized language for encoding and communicating information about cyber observables ([http://cybox.mitre.org](http://cybox.mitre.org))
A Brief History of Cyber Observables

- September 2009: Concept introduced to CAPEC in Version 1.4 as future envisioned adornment to the structured Attack Execution Flow
- June 2010: Broader relevance to MSM recognized leading to CAPEC, MAEC & CEE teams collaborating to define one common structure to serve the common needs
- August 2010: Discussed with US-CERT at GFIRST 2010
- December 2010: Cyber Observables schema draft v0.4 completed
- December 2010: Discussions with Mandiant for collaboration and alignment between Cyber Observables and Mandiant OpenIOC
- January 2011: Discussed & briefed with MITRE CSOC
- February 2011: Discussed & briefed with NIST – EMAP and US-CERT who also have a need for this construct and had begun to work on parallel solutions
Cyber Observables Apply to Numerous Domains

- Threat assessment & characterization (detailed attack patterns)
- Malware characterization
- Operational event management
- Logging
- Cyber situational awareness
- Incident response
- Forensics
- Etc.

Through utilization of the standardized CybOX language within the relevant standards, incident response and management can take advantage of all of these aligned capabilities to investigate occurring incidents, improve overall situational awareness and improve future attack detection, prevention and response.
CybOX Principles

■ Targeted Gestalt
  – CybOX is not targeted at a single cyber security use case but rather is intended to be flexible enough to offer a common solution for all cyber security use cases requiring the ability to deal with cyber observables.

■ Flexibility to express both instances and apriori potential patterns
  – It is also intended to be flexible enough to allow both the high-fidelity description of instances of cyber observables that have been measured in an operational context as well as more abstract patterns for potential observables that may be targets for observation and analysis apriori.

■ Integrated automation vision
  – By specifying a common structured schematic mechanism for these cyber observables, the intent is to enable the potential for detailed automatable sharing, mapping, detection and analysis heuristics.
Cyber Observable Broader Use Cases

- Detect malicious activity from attack patterns
- Empower & guide incident management
- Identify new attack patterns
- Prioritize existing attack patterns based on tactical reality
- Potential ability to analyze data from all types of tools and all vendors
- Improved sharing among all cyber observable stakeholders
- Ability to metatag cyber observables for implicit sharing controls
- Enable automated signature rule generation
- Enable new levels of meta-analysis on operational cyber observables
- Potential ability to automatically apply mitigations specified in attack patterns
- Etc....
CybOX adorned attack patterns have the potential for being a very important bridge between the two domains, as they enable the alignment of the low-level aggregate mapping of observables that occurs in the operations domain to the higher-level abstractions of attacker methodology, motivation, and capability that exist in the development domain.
Industry Uptake

Testing

Fuzz testing is a reliability and security testing technique that relies on building intentionally malformed data and then having the software under test consume the malformed data to see how it responds. The science of fuzz testing is somewhat new but it is maturing rapidly. There is a small market for fuzz testing tools today, but in many cases software developers must build bespoke fuzz testers to suit specialized file and network data formats. Fuzz testing is an effective testing technique because it uncovers weaknesses in data handling code.

CWE CAPEC

2. The MITRE Corporation, Common Weakness Enumeration (CWE), http://cwe.mitre.org/
Linkage with Fundamental Changes in Enterprise Security Initiatives

Twenty Critical Controls for Effective Cyber Defense: Consensus Audit Guidelines

What the 20 CSC Critics say...

20 Critical Security Controls - Version 2.0

- 20 Critical Security Controls - Introduction (Version 2.0)
- Critical Control 1: Inventory of Authorized and Unauthorized Devices
- Critical Control 2: Inventory of Authorized and Unauthorized Software
- Critical Control 3: Secure Configurations for Hardware and Software
- Critical Control 4: Secure Configurations for Network Devices
- Critical Control 5: Boundary Defense
- Critical Control 6: Maintenance, Monitoring, and Analysis of Information Systems
- Critical Control 7: Application Software Security
- Critical Control 8: Controlled Use of Administrative Privileges
- Critical Control 9: Controlled Access Based on Need to Know
- Critical Control 10: User Access Management
- Critical Control 11: Asset Management
- Critical Control 12: Incident Response
- Critical Control 13: Risk Management
- Critical Control 14: Security Assessments
- Critical Control 15: Automated Vulnerability Scanning
- Critical Control 16: Vulnerability Management
- Critical Control 17: Configuration Management
- Critical Control 18: Security сказать...
- Critical Control 19: Access Control
- Critical Control 20: Data Protection

CWE and CAPEC included in Control 7 of the “Twenty Critical Controls for Effective Cyber Defense: Consensus Audit Guidelines”
Common Criteria v4 CCDB
- TOE to leverage CAPEC & CWE
- Also investigating how to leverage ISO/IEC 15026 NIAP Evaluation Scheme
- Above plus
- Also investigating how to leverage SCAP
CAPEC Future Plans

• V1.7 (within the next week or two)
  • Will flesh out ~30-40 stub patterns to full patterns
  • Will add 15-20 new stub patterns
  • Will include existing content that has been refined for quality & consistency
  • Will align to a common consequences & technical impacts structure between CWE and CAPEC
  • Establish initial compatibility program

• Strategic focus for the near to mid-term will be on utilizing CAPEC as a bridge between secure development and secure operations

• Continue expanding and refining content

• Continue expanding outreach and supporting CAPEC use
The HS SEDI FFRDC is managed and operated by The MITRE Corporation for DHS.

### Enterprise IT Asset Management
- **Asset Inventory**
  - CPE/ OVAL/ ARF
- **Configuration Guidance Analysis**
  - CCE/ CCSS/ OVAL/ARF/ XCCDF/CPE
- **Vulnerability Analysis**
  - CVE/CWE/ CVSS/ARF/ CCE/CCSS/ ARF/CWSS/ OVAL/CPE/ XCCDF
- **Threat Analysis**
  - CVE/CWE/ CVSS/ARF/ CCE/CCSS/ ARF/CWSS/ XCCDF/CPE/ CAPEC/MAEC
- **Intrusion Detection**
  - CVE/CWE/ CVSS/ARF/ CCE/CCSS/ OVAL/CWSS/ XCCDF/CPE/ CAPEC/MAEC
- **Incident Management**
  - CVE/CWE/ CVSS/ARF/ CCE/CCSS/ OVAL/ARF/ XCCDF/CPE

### Operations Security Management Processes
- **Centralized Reporting**
- **Development & Sustainment Security Management Processes**
- **Assessment of System Development, Integration, & Sustainment Activities and Certification & Accreditation**
  - CWE/CAPEC/ CWSS/MAEC/ OVAL/OCIL/X CCDF/CCE/CP E/ARF/SAFES/ SACM

### Operational Enterprise Networks
- **INTERNET**
  - Router
- **DMZ**
  - Firewall
  - Web Servers
  - Application Servers
  - Database Systems
  - Desktop Systems

### Trust Management
- **Centralized Reporting**
- **Enterprise IT Change Management**
- **Identity Management**
- **Homeland Security**

The HS SEDI FFRDC is managed and operated by The MITRE Corporation for DHS.
The HS SEDI FFRC is managed and operated by The MITRE Corporation for DHS.

- **Asset Inventory**
- **Configuration Guidance Analysis**
- **Vulnerability Analysis**
- **Threat Analysis**
- **Intrusion Detection**
- **Incident Management**

**Operations Security Management Processes**

**Development & Sustainment Security Management Processes**

**Trust Management**

**Enterprise IT Change Management**

**Identity Management**

**Centralized Reporting**

**Operational Enterprise Networks**

**Asset Inventory**

**Configuration Guidance Analysis**

**Vulnerability Analysis**

**Threat Analysis**

**Incident Management**

**Operations Security Management Processes**

**Development & Sustainment Security Management Processes**

**Trust Management**

**Enterprise IT Change Management**

**Identity Management**

**Centralized Reporting**

**Operational Enterprise Networks**

CPE/CAPEC/CWSS/MAEC/CVSS/CCSS/OVAL/XCCDF/CPE/ARF/SACM
The HS SEDI EFRC is managed and operated by The MITRE Corporation for DHS.

Asset Inventory
Configuration Guidance Analysis
Vulnerability Analysis
Threat Analysis
Intrusion Detection
Incident Management

Mitigating Risk Exposures
Responding to Security Threats

Operations Security Management Processes

Development & Sustainment Security Management Processes

System & Software Assurance Guidance/Requirements

Assessment of System Development, Integration, & Sustainment Activities and Certification & Accreditation

Trust Management
Enterprise IT Change Management
Identity Management
Centralized Reporting

Operational Enterprise Networks

The HS SEDI EFRC is managed and operated by The MITRE Corporation for DHS.

Enterprise IT Asset Management
Knowledge Repositories

- Asset Inventory: CPE/OVAL
- Configuration Guidance Analysis: CPE/OVAL/CCE/CCSS
- Vulnerability Analysis: CVE/CVE/OVAL/CVSS
- Threat Analysis: CVE/CVE/CVSS
- Intrusion Detection: CAIF/IDMEF/IODEF/CVE/CVE/CVE/CWE/CPE/MAEC
- Incident Report: CVE/CVE/CVSS/CPE/CWSS/CCE/ARF

System & Software Assurance Guidance/Requirements

- Assessment of System Development, Integration, & Sustainment Activities and Certification & Accreditation: CPE/CAPEC/SBVR/CWSS/MAEC

Operations Security Management Processes

- Operations Enterprise Networks: CVE/CWE/CVSS/CCE/CCSS/OVAL/XCCDF/CPE/CAPEC/MAEC/CWSS/CEE/ARF

Centralized Reporting
Knowledge Repositories

- Asset Definition
- Configuration Guidance
- Vulnerability Alert
- Threat Alert
- Intrusion Detection
- Incident Report

Assessment of System Development, Integration, & Sustainment Activities and Certification & Accreditation

System & Software Assurance Guidance/Requirements

Operations Security Management Processes

Internet

Router

Web Servers

Application Servers

Database Systems

DMZ

Firewall

Desktop Systems

Desktop Systems

Desktop Systems

Operational Enterprise Networks

Development & Sustainment Security Management Processes

Trust Management

Enterprise IT Change Management

Identity Management

Centralized Reporting

The HS SEDI FFRC is managed and operated by The MITRE Corporation for DHS.
The HS SEDI FRDC is managed and operated by The MITRE Corporation for DHS.

Knowledge Repositories

Asset Inventory

Configuration Guidance Analysis

Vulnerability Analysis

Threat Analysis

Intrusion Detection

Incident Report

Asset Definition

Configuration Guidance

Vulnerability Alert

Threat Alert

Centralized Reporting

Operations Security Management Processes

Assessment of System Development, Integration, & Sustainment Activities and Certification & Accreditation

System & Software Assurance Guidance/Requirements

SAFES

SAFES

Development & Sustainment Security Management Processes

Trust Management

Enterprise IT Change Management

Identity Management

Centralized Reporting

Operational Enterprise Networks

Integration

Router

Web Servers

Application Servers

Database Systems

Network Security Management Processes

SAFES

SAFES

SAFES

SAFES

SAFES

CPE/OVAL

CPE/OVAL/ARF

CCE/CCSS/ARF/CPE/XCCDF

CVE/CVE/CSS/ARF/CCE/CCSS/OVAL/CWSS/XCCDF/CPE/CAPEC/MAECC

CIF/IDMIF/ID/EE/CIF/CVE/CA/CPE/CAEC/CCSS/CWSS/CEE/CE/ARF

OVAL/XCCDF/CCCE/CCE/CESS/CPE/ARF

CWE/CAPE/CWSS/MAEC/OVAL/OOIL/XXDF/CCE/CPE/CAPE/ARF/SAFECS/SAFECS

CVE/CWE/CVSS/CE/CSS/OVAL/XCCDF/CPE/CAPE/CWSS/CE/ARF
Knowledge Repositories

- Asset Inventory
- Configuration Guidance
- Vulnerability Analysis
- Threat Analysis
- Intrusion Detection
- Incident Report

Operations Security Management Processes

- System & Software Assurance Guidance/Requirements
- SACM

Assessment of System Development, Integration, & Sustainment Activities and Certification & Accreditation

Development & Sustainment Security Management Processes

- Trust Management
- Enterprise IT Change Management
- Identity Management
- Centralized Reporting

Operational Enterprise Networks

- Web Servers
- Application Servers
- Database Systems
- Desktop Systems

The HS SEDI FRDC is managed and operated by The MITRE Corporation for DHS.
Questions?

sbarnum@mitre.org